Theoretical Computer Science 61 (1988) 199-224
North-Holland

THE EQUATIONAL THEORY OF POMSETS

Jay L. GISCHER
Department of Computer Science, College of William and Mary, Williamsburg, VA 23185, U.S.A.

Communicated by T. Ito
Received September 1986
Revised June 1987

Abstract. Pomsets have been introduced as a model of concurrency. Since a pomset is a string in which the total order has been relaxed to be a partial order, in this paper we view them as a generalization of strings, and investigate their algebraic properties. In particular, we investigate the axiomatic properties of pomsets, sets of pomsets and ideals of pomsets, under such operations as concatenation, parallel composition, union and their associated closure operations. We find that the equational theory of sets, pomsets under concatenation, parallel composition and union is finitely axiomatizable, whereas the theory of languages under the analogous operations is not. A similar result is obtained for ideals of pomsets, which incorporate the notion of subsumption which is also known as augmentation. Finally, we show that the addition of any closure operation (parallel or serial) leads to nonfinite axiomatizability of the resulting equational theory.

1. Introduction

Recently, many researchers have turned to partial orders in order to aid in understanding the semantics of parallel programs. Consider the work of Lamport [18, 19], and Winskel [30, 31] as just two examples of this trend. The advocates of net theory have long held this point of view [3, 9]. The particular formalism which is inspired by partial orders that we concern ourselves with is known as the partially ordered multiset or pomset. These were first suggested as a solution to what has become known as the Brock-Ackerman anomaly [4] to the dataflow networks of Kahn and MacQueen [16]. Pratt has written many papers showing how pomsets can be used to represent parallel processes of a very general nature and how they can be composed [23, 24].

This work was carried out independently from that of Grabowski [9], who introduced the notion of a partial string and a partial language, which coincide exactly with the notions of pomset and process in this paper. Grabowski was concerned with characterizing the notion of firability of a partial string by a Petri net. His paper explores the relationship between partial strings and Petri nets, as well as discussing the notion of Y-product, which is closely related to Pratt's process composition. We hope that the reader will indulge us if we, bowing to habit, use the terms "pomset" and "process" where Grabowski used "partial string" and "partial language". What distinguishes our work from Grabowski's is the logical

flavor of our work. We develop the equational theory of a selected set of operations on processes.

Milner has done similar work with *behaviors* [20, 21], applying a technique due to Salomaa to these objects which resemble trees of possible nondeterministic execution. What distinguishes our work from Milner lies in the differences between behaviors and pomsets. Behaviors are a two-typed model, in the sense that there are atomic actions and (possibly) nonatomic behaviors. Furthermore, sequencing of events is only done by prefixing a behavior with an atomic event. In this setting, concurrency is equivalent to nondeterministic interleaving. One main thrust of pomsets is to provide a model in which there is a notion of concurrency which is independent of nondeterminism (although by no means unrelated to it). In the pomset world there is only one type of object, which can be treated as atomic or nonatomic as the need arises.

Another difference, although rather technical, is that Milner's axiomatization is not really an equational aximatization since an additional rule of inference is added to the usual ones for equations. However, this is probably necessary to get completeness for the fixed-point operation, which is probably no more finitely axiomatizable in his model than it is for regular algebras or pomsets.

Very similar to Milner's work on behaviors is Ito and Ando's work on superregular expressions [12]. Nearly everything we said about Milner's inference system applies to Ito and Ando's work as well; Milner's work could be seen as an extension of Ito's.

Another line of research that is relevant to this paper is the work done on propositional dynamic logic (or PDL) done by various researchers. While the present paper would seem to have little to do with PDL, one of the motivations for studying the axiomatic properties of pomsets is related to our wish to develop a dynamic logic of parallel processes. Some work has been done in this area by Abrahamson [1] using formal language theory employing the traditional shuffle operation. Based on some of the results in Section 6, we believe that such "total-order" semantics are inadequate to express the true richness of processes, hence we would like to develop a dynamic logic of pomsets. But in doing so, it would be of great value if we knew the axiomatic properties of some of the more basic pomset operations.

A final strand of research that is relevant is the work on the axiomatic theory of regular algebra done by Redko [25], Salomaa [26], and Conway [6]. Much of the inspiration for this work is taken from their publications, especially Conway's book.

A good deal of research has been done categorizing various flavors of shuffle expressions, where the variables are interpreted as symbols in an alphabet rather than as languages, so that all objects are in this sense atomic. Such work includes that of Jantzen [14, 15], Slutzki [28], Gischer [7], Shaw [27], Kimura [17], Araki et al. [2] and others. Because of the implied atomicity of such languages, this work is of interest to readers of this paper, but is not directly relevant.

What is a pomset anyway? The notion is that we want to have a trace of a process; by a trace we mean some sort of sequencing of activities which we call *tasks*. A task can be thought of as some activity, which can be performed several times and has

a duration. Thus there are two important distinctions between tasks and *actions* which are usually thought to be atomic (i.e., taking effect instantaneously) and unique. One consequence of tasks having duration in time is that they may overlap in time. Thus we cannot assign sequences to them; neither occurs before the other. In consequence of the repeatability of tasks, we see that any trace will have to allow repeated symbols. But this brings up a tricky technical point: How do we say in our formal model that one occurrence of task A precedes another occurrence of task A while still another occurrence of A overlaps in time with both the previous occurrences of A. It would seem that we need some way to temporarily distinguish between different occurrences of the same task.

Our solution is to first define an underlying partial order of distinct vertices, and then to label them with the tasks we have in mind for each of them. Finally, we abstract away the actual identity of the underlying partial order, leaving only its structure. More formally, a *labelled partial order* is a 4-tuple $(V, \Sigma, <, \mu)$ where V is a set of vertices, $<$ is a partial ordering of V, Σ is an alphabet and μ maps V to Σ. Thus V and $<$ provide the underlying partial ordering, while μ labels each vertex with the name of some task, i.e., a symbol from our alphabet Σ.

Let $P=(V_P, \Sigma_P, <_P, \mu_P)$ and $Q=(V_Q, \Sigma_Q, <_Q, \mu_Q)$ be labelled partial orders. We say that P and Q are *isomorphic* if and only if there is a mapping from V_P to V_Q which preserves ordering and labelling. That is, there is some τ for which $\mu_Q(\tau(v))=\mu_P(v)$ and $u<_P w$ iff $\tau(u)<_Q\tau(w)$. A *partially-ordered multiset*, or pomset, is the isomorphism class of some labelled partial order. For a given labelled partial order such as P, we denote its corresponding pomset by using square brackets, e.g., $[V_P, \Sigma_P, <_P, \mu_P]$ is the pomset corresponding to P. In this way we can now ignore what the actual contents of V_P are and content ourselves with dealing with an archetypal representative of any pomset, namely the one given inside the brackets.

We can now define other familiar set-theoretic constructs in terms of pomsets. A *tomset*, or totally-ordered multiset, is none other than a string. A *multiset* is a pomset with the empty partial order on its vertices. A *set* is a multiset with an injective labelling function. With the preceding definitions, we can deal with many varied set-theoretic structures in a uniform manner.

2. Substitution, homomorphism, and pomset-definable operations

We have adopted a very general notion of task by not assuming our tasks to be atomic. In particular, we wish to leave open the possibility of "looking inside" some task to see its components, which are themselves represented as pomsets. The formal notion of this operation is what we call a substitution.

Let $J=[V_J, \Sigma_J, <_J, \mu_J]$ be a pomset and let g be a function mapping each a in Σ_J to some pomset, which we will denote by $g(a)=[V_{g(a)}, \Sigma_{g(a)}, >_{g(a)}, \mu_{g(a)}]$. We can think of g as an assignment to variables, associating with each symbol of Σ_J a corresponding pomset. Then the *substitution* $J[g]$ is a new pomset $[V_{J[g]}, \Sigma_{J[g]},$

$<_{J[g]}, \mu_{J[g]}]$, in which each instance of a symbol a is replaced with an instance of its corresponding pomset $g(a)$, and if v_1 was less than v_2 in the original, then all the nodes of the new pomset corresponding to v_1 ($g(\mu_J(v_1))$) are made to be less than all the nodes of the new pomset corresponding to v_2. See Fig. 1.

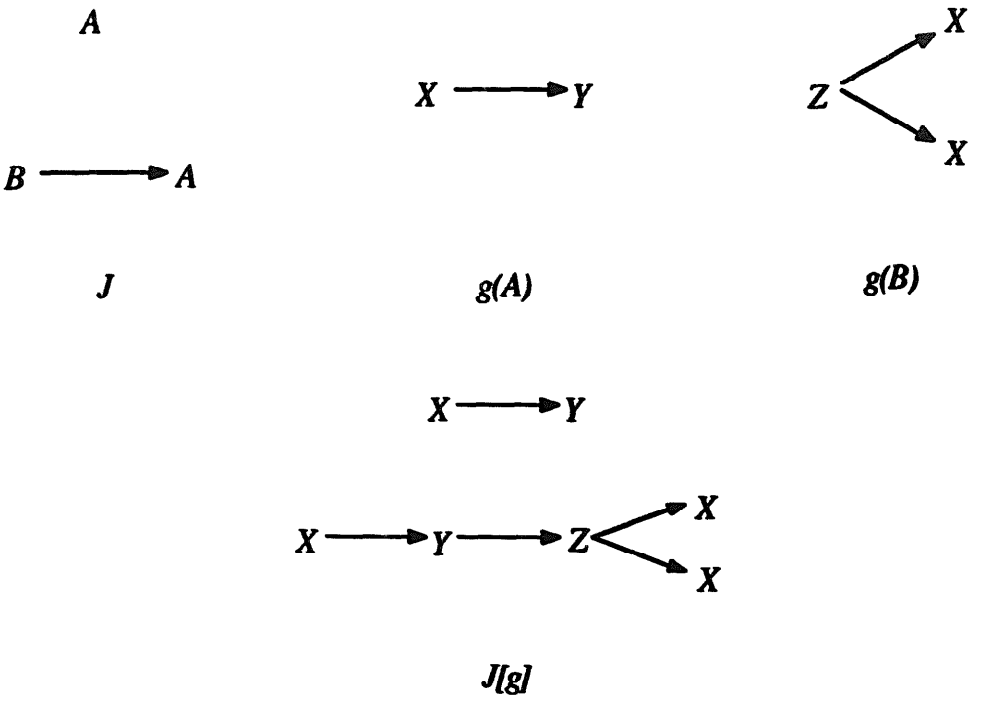

Fig. 1. Substitution.

We now give a more rigorous definition of $J[g]$. By composing μ_J with g we get a function f which associates a pomset with each vertex of J. Then $V_{J[g]}$ is the disjoint union of all the $V_{f(v)}$ for all v in V_J. Likewise, $\Sigma_{J[g]}$ is the union of $\Sigma_{f(v)}$ for all $v \in V_J$. The labelling function of $J[g]$ is defined by making, for each node w in $V_{J[g]}$, $\mu_{J[g]}(w)$ the label that w was labelled with in the pomset from whence it came. Finally, the relation $w_1 <_{J[g]} w_2$ holds if and only if either $w_1 <_{f(v)} w_2$ for some $v \in V_J$, or else $w_1 \in V_{f(v_1)}$, $w_2 \in V_{f(v_2)}$ and $v_1 <_J v_2$. We summarize with the following formulas for $J[g]$, letting $f = g \circ \mu_J$:

$$V_{J[g]} = \biguplus_{v \in V_J} V_{f(v)}, \qquad \Sigma_{J[g]} = \bigcup_{v \in V_J} \Sigma_{f(v)},$$

$$<_{J[g]} = \left(\bigcup_{v_1 <_J v_2} V_{f(v_1)} \times V_{f(v_2)} \right) \cup \left(\bigcup_{v \in V_J} <_{f(v)} \right),$$

$$\mu_{J[g]} = \biguplus_{v \in V_J} \mu_{f(v)}.$$

Now suppose that we fix a particular function g_0 mapping some alphabet Σ to pomsets, and let the variable J range over all pomsets with alphabet Σ in the expression $J[g_0]$. What results is a unary operation $\lambda J.J[g_0]$ mapping pomsets to

pomsets, or a *pomset homomorphism*, so named because of its similarity to language homomorphisms. Pomset homomorphisms are what we have in mind when we speak of "revealing internal structure"; a single symbol is replaced by a more detailed pomset.

We can generalize pomset homomorphisms even further by allowing an "assignment" function $\bar{g}$ which maps Σ to sets of pomsets rather than individual pomsets. If $\bar{g}$ is such a function, then say that h is *compatible* with $\bar{g}$ if $h(a) \in \bar{g}(a)$ for all $a \in \Sigma$. Then we extend the meaning of substitution to make

$$J[\bar{g}] = \bigcup_{h \in C(\bar{g})} J[h],$$

where $C(\bar{g})$ denotes all functions compatible with $\bar{g}$.

Instead of holding g constant in $J[g]$, we could hold J constant instead, the result being a unary operation that takes an "assignment" function g and produces another pomset, i.e., $\lambda g.J_0[g]$. This we refer to as a pomset-definable operation. We may think of the function g as being a tuple of arguments, or a binding of formal parameters to actual parameters, which suggests the following notation. If we let $\tilde{A} = (a_1, \ldots, a_m)$ be an element of Σ^m such that each a_i is unique, and let $\tilde{J} = (J_1, \ldots, J_m)$ be a tuple of pomsets, then we shall let a/J denote the assignment which maps a to J and leaves all other points fixed, whereas $\tilde{A}/\tilde{J}$ will denote the function which maps a_i to J_i for all i between 1 and m.

3. A characterization of series-parallel pomsets

The fundamental operation on strings is that of concatenation, which, from strings x and y, produces the string $x.y$, usually written just xy, which consists of x followed by y. This operation can be extended to pomsets in the following way: Given pomsets J and K, produce the concatenation $J.K$ by taking the disjoint sum of J and K and making all the vertices of J precede all those of K according to $<_{J.K}$. The reader should note that this new notion of concatenation coincides exactly with the old one when the objects operated on are tomsets (strings). The manner of our construction perhaps suggests another operation which we can apply to pomsets, the operation which combines J and K in a disjoint sum, adding no further edges between J and K. We call this operation the *concurrent composition* of J and K and denote it by $J \| K$. For those who find "concurrent composition" too verbose, we suggest that $J \| K$ be pronounced "J co K."

We observe in passing that both of the above operations are pomset-definable. In fact, they correspond to the only two "shapes" possible for pomsets of two elements. Concatenation and concurrent composition correspond nicely to the programming concepts of sequential and parallel composition (i.e. ";" and "cobegin-coend") and it is natural to turn our attention to exactly those pomsets which can be constructed from single points by the operations of concatenation and concurrent composition.

Let J be known as a *unit pomset*, more informally, a unit, just when V_J consists of a single element. In the future, we will make no distinction between the unit pomset and the symbol labelling it since there is an obvious correspondence. The *series-parallel* pomsets is the smallest set of pomsets that includes both the unit pomsets and the empty pomset and is closed under the operations . and ||. Now we will give an important characterization of series-parallel pomsets, first due to Grabowski [9]. This characterization was independently discovered by Valdes, and reported in [29], where it was described in terms of vertex series-parallel graphs.

We define pomsets J and K to be *similar* just when $<_J$ and $<_K$ are isomorphic. Similarity is intended to capture the notion of two pomsets having the same "shape" much as, in geometry, similar triangles have the same shape. We say that K is a *subpomset* of J just when $V_K \subset V_J$, $\Sigma_K = \Sigma_J$ and $<_K$ and μ_K result from restricting the domains of $<_J$ and μ_J respectively. There is no requirement that the vertices of the subpomset be adjacent, so that *ac* is a subpomset of *abcd*. Last but not least we define the pomset $N = [V_N, \{a\}, <_N, \mu_N]$ as follows:

$$V_N = \{1, 2, 3, 4\}, \qquad <_N = \{(1, 3), (2, 4), (1, 4)\},$$
$$\mu_N(i) = a \quad \text{for all } i \in V_N.$$

The pomset N gets its name from its characteristic shape which looks like an "N". Any pomset which has a subpomset which is similar to N is said to be *N-full* or to contain an "N". A pomset which has no subset similar to N is said to be *N-free*. We are now ready to state the characterization of series-parallel pomsets.

Theorem 3.1. *A finite pomset is series-parallel if and only if it is N-free.*

Proof. This is proved for vertex-series parallel graphs by Valdes in [29], and for partial strings by Grabowski in [9]. However, we give our own proof here as it will give some insight that will be useful later on, as well as being substantially shorter than either of the above proofs.

Let J be a finite pomset. First, suppose that J is series-parallel. Then an easy structural induction proof shows that J is N-free. For if J is a unit, then it is N-free trivially. If J is the concurrent composition of J_1 and J_2, then the induction hypothesis gives us that J_1 and J_2 are N-free, but since J contains no new edges, it too must be N-free. Finally, if $J = J_1.J_2$, then J_1 and J_2 are N-free by the induction hypothesis. Now suppose we have vertices u, v, x, and y forming an N, i.e., $u < x$, $v < y$, and $u < y$ and no other relations. At least one of them must be a vertex of J_1 and likewise for J_2, lest N-freeness of these pomsets be violated. But then u and v must be vertices of J_1, and x and y vertices of J_2, else one vertex will be a minimal (or maximal) element, which would not give an N. But this means that $v < x$, which is a contradiction. We conclude that J must be N-free.

Now suppose that J is N-free. We will show that there is a unique decomposition of J as a series-parallel pomset if we ignore associativity of . and || and commutativity of ||. We proceed by induction on the number of vertices of J. The basis case occurs when J is a unit, which is both series-parallel and N-free.

In the induction step, there are two cases: Either J is connected or else it is not. First suppose J is not connected. Then let $J_1, \ldots, J_n$ be the set of all the connected components of J. It follows that $J = J_1 \| \cdots \| J_n$. Each of the J_i's must be N-free, so it follows that they are series-parallel from the induction hypothesis, each with a unique parsing. Unique parsing of J follows immediately.

Next suppose that J is connected. We divide J into two subpomsets J_1 and J_2 such that $J = J_1.J_2$ and J_1 is not connected. Let M be the set of minimal elements of J. If $|M| = 1$, then let J_1 be M and J_2 be $J - M$. Let D be the set of all vertices, each of which is greater than every element of M. We claim that D is nonempty. If not, then there is a minimal element u and a maximal element x such that $u \not< x$. Since J is connected, there is a shortest sequence of vertices u_i, x_i such that $u < x_1 > u_1 < x_2 > \cdots > u_n < x$, with each u_i minimal and each x_i maximal. But since J is N-free, we have $u < x_2$, contradicting minimality of the sequence. So all the maximal elements of J must be in D, and D must be nonempty.

We claim that each element of D is greater than every element of $J - D$, i.e., $J = (J - D).D$. Suppose not. Then we have $d \in D$ and $e \in J - D$ with d and e unrelated. ($e > d$ would imply $e \in D$.) Now e cannot be a minimal element, so let $m < e$ be a minimal element. Since $e \notin D$, there is a minimal element n with $n \not< e$. Thus d, e, m, and n form an N, contradicting N-freeness of J (see Fig. 2). Thus $J = J_1.J_2$. By the induction hypothesis, J_1 and J_2 have unique parsings as series-parallel graphs. If J_2 is connected, the above process is repeated until we have a sequence $J_1, \ldots, J_m$ such that each J_i is not connected and $J = J_1 \ldots J_m$. This decomposition is unique up to associativity. So J is series-parallel, proving the theorem. □

To reiterate, in the above theorem, we have shown that every finite N-free pomset has a unique (when ignoring associativity and commutativity) decomposition as a series-parallel pomset.

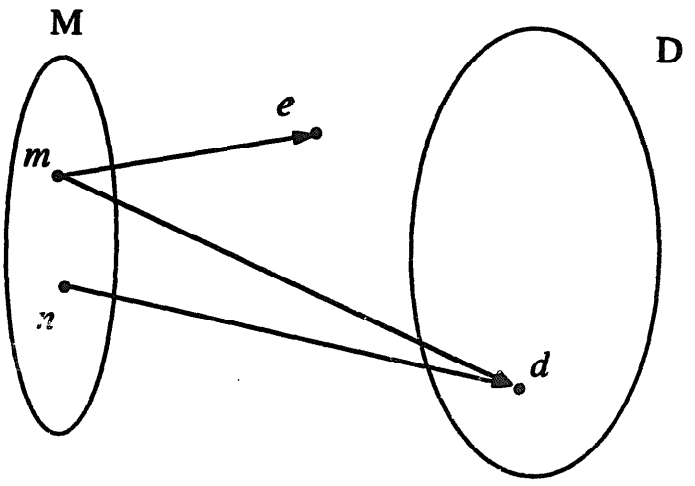

Fig. 2. There must be an edge from e to d.

Every series-parallel pomset is, of course, finite. The above characterization suggests that if we wish to consider infinite pomsets which have a similar structured nature, we should look at the N-free pomsets.

Series-parallel pomsets are of critical importance. Every term over the operations . and $\|$ can be thought of as defining an operation on pomsets in the obvious way. But that operation is of necessity a pomset-definable operation, and furthermore, the pomset defining it is series-parallel. So our interest in series-parallel pomsets is due partly to this correspondence. Notationally, we refer to the pomset corresponding to the term t as $\rho(t)$. Observe that ρ is not one-to-one. For example, $\rho(x\|y) = \rho(y\|x)$. It is then the kernel of ρ that is of interest to us; that is, if $\rho(t) = \rho(r)$, then we will think of t and r as being equivalent and we will write $t \equiv r$.

We now present two lemmas which will be of use later on. We say that a pomset J is *prime* if there do not exist nonempty pomsets K_1 and K_2 such that $J = K_1.K_2$. The next lemma was originally proved by Graham, Knuth and Motzkin [32] about posets, but applies equally well to pomsets.

Lemma 3.2. *Every pomset has a unique factorization (ignoring associativity of concatenation) into primes.*

Proof. A simple adaptation of the proof of the previous theorem will suffice here and is left as an exercise to the reader. □

We conclude this section with a lemma which we will find useful later on. Let $J = [V_J, \Sigma_J, <_J, \mu_J]$ be a pomset. For $u, v \in V_J$, define $d(u, v)$ to be the length of the shortest path between u and v, ignoring orientation of edges. If no such path exists, then let $d(u, v) = 0$. Now define $D(J)$ to be the maximum value of $d(u, v)$ over all choices of u and v in V_J. We call $D(J)$ the *diameter* of J. If we let N be as above, we can see that $D(N) = 3$, whereas if we remove any edge from $<_N$ to get N', we get $D(N') \leq 2$. This observation leads to the following theorem.

Theorem 3.3. *Let J be a pomset. Then J is N-free if and only if $D(J) \leq 2$.*

Proof. If $D(J) > 2$, then there exist $u, v \in V_J$ such that $d(u, v) = 3$. But then J contains a subpomset similar to N.

If J is N-free, then J cannot contain a subpomset similar to N. But then we must have $D(J) \leq 2$. □

4. Axiomatic questions

The motivation for introducing pomsets is to define a *process* as a set of pomsets. In other words, a process is identical to the set of its possible behaviors. In this

regard the operation ., also known to some authors as ";", is sequential composition, and $\|$ is concurrent composition. Finally, choice or variety of action, whether deterministic or not, is represented as the operation +.

One possible interpretation of these operations, assuming that P and Q are processes is as follows:

$$P.Q = \{J.K \mid J \in P \wedge K \in Q\},$$

$$P \| Q = \{J \| K \mid J \in P \wedge K \in Q\}, \qquad P + Q = P \cup Q.$$

In this section we concern ourselves with the equivalence of terms under the above operations as well as some derived operations. Another way of saying the same thing is that we are concerned with the equational theory of the operations concatenation, concurrency, and choice. The equational theory is by no means the whole story, but is a necessary step toward developing the theory of, say, dynamic logic of pomsets.

First, let us consider the operations of . and $\|$ without choice. We will say that terms t and r are S-equivalent ($t \equiv_S r$) iff t and r always yield the same process whenever their variables are interpreted as processes.

Lemma 4.1. *If t and u are terms over the operations . and $\|$, then $t \equiv_S u$ if and only if $\rho(t) = \rho(u)$.*

Proof. Let t be a term over the variables in the tuple $\tilde{X} = (X_1, \ldots, X_n)$ and let $\tilde{Y}$ be the tuple $\tilde{Y} = (\{X_1\}, \ldots, \{X_n\})$. First observe that $\{\rho(t)\} = t[\tilde{X}/\tilde{Y}]$. Thus $t \equiv_S u$ implies that

$$\{\rho(t)\} = t[\tilde{X}/\tilde{Y}] = u[\tilde{X}/\tilde{Y}] = \{\rho(u)\}$$

which implies that $\rho(t) = \rho(u)$.

Next, observe that, for any substitution g, $t[g] = \rho(t)[g]$; hence, if $\rho(t) = \rho(u)$, then it follows that $t[g] = \rho(t)[g] = \rho(u)[g] = u[g]$ for any substitution g, i.e., $t \equiv_S u$. □

There are some laws that follow immediately from the definitions above. First of all, both operations are associative, and $\|$ is commutative.

$$P.(Q.R) \equiv_S (P.Q).R, \tag{1}$$

$$P \| (Q \| R) \equiv_S (P \| Q) \| R, \tag{2}$$

$$P \| Q \equiv_S Q \| P. \tag{3}$$

If we let 1 denote the process which contains only the empty pomset, and 0 the empty process, we have the following laws:

$$P.1 \equiv_S 1.P \equiv_S P, \qquad P \| 1 \equiv_S P, \tag{4, 5}$$

$$P.0 \equiv_S 0.P \equiv_S 0, \qquad P \| 0 \equiv_S 0. \tag{6, 7}$$

We take the soundness of equations (1)-(7) to be self-evident. What is not so obvious is that they are *complete* for processes as well. That is to say, suppose t_1 and t_2 are terms over the operations . and $\|$, with the variables to be interpreted as processes. Then if $t_1 \equiv_S t_2$, it is possible to deduce that fact using the above equations and the rules of high-school algebra. We summarize in the following theorem.

Theorem 4.2. *For all terms t and u over the operations . and $\|$, t is S-equivalent to r if and only if $t \equiv_S u$ can be proved from equations* (1)-(7) *above.*

Proof. Suppose $t \equiv_S r$. That means that $\rho(t) = \rho(r)$. If either t or r contains the symbol "1" or "0", then there are terms t' and r' respectively which do not contain "0" or "1" such that $t \equiv_S t'$ and $r \equiv_S r'$, or else $t' = r' = 0$. Furthermore, $t \equiv_S t'$ and $r \equiv_S r'$ are provable using axioms (4)-(7). Therefore, in what follows, we assume that neither "0" nor "1" occurs in either of t or r. Now $\rho(t)$ is series-parallel. We proceed by induction on the size of $\rho(t)$.

Case 1. Let $\rho(t)$ consist of a single vertex. Then it follows that t, and hence r, are variables, and $t = r$.

Case 2. Let $\rho(t)$ be connected. Then $\rho(t)$ is $T_1 \ldots T_n$, for some pomsets $T_1, \ldots, T_n$ such that each T_i is not connected. This follows from the unique parsability of $\rho(t)$ shown in Theorem 3.1. It follows that there are terms $t_1, \ldots, t_n$ and $r_1, \ldots, r_n$ such that $\rho(t_i) = \rho(r_i) = T_i$ for all i, $1 \leq i \leq n$, and such that t is the term $t_1 \ldots t_n$ associated in some arbitrary fashion and likewise r is the term $r_1 \ldots r_n$ associated in some fashion. Now, by the induction hypothesis, each of the equations $t_i \equiv_S r_i$ is provable from axioms (1)-(7). But then $t \equiv_S r$ follows from the associativity of concatenation.

Case 3. Let $\rho(t)$ have more than one connected component. Then $\rho(t)$ is $T_1 \| \cdots \| T_n$ for some pomsets $T_1, \ldots, T_n$ such that each T_i is connected. It then follows that there are terms $t_1, \ldots, t_n$ and $r_1, \ldots, r_n$ such that $\rho(t_i) = \rho(r_i) = T_i$ for all i, $1 \leq i \leq n$, and such that t is the term $t_1 \| \cdots \| t_n$ associated in some arbitrary fashion and such that there is a permutation $p(i)$ on $\{1, \ldots, n\}$ for which r is the term $r_{p(1)} \| \cdots \| r_{p(n)}$ associated in some manner. By the induction hypothesis, each of the equations $t_i \equiv_S r_i$ is provable from the axioms. But then $t \equiv_S r$ follows from the associativity and commutativity of concurrent composition. This completes the induction proof.

The other direction amounts to showing the soundness of the axioms, which can be easily verified. □

Should we wish to add choice (+) to our vocabulary of operations, the situation is not a great deal more complicated. In addition to equations (1)-(7) above, we have the following axioms. First, choice is associative, commutative, and idempotent, with 0 as its identity.

$$P + (Q + R) \equiv_S (P + Q) + R, \qquad P + Q \equiv_S Q + P, \tag{8, 9}$$

$$P + 0 \equiv_S P, \qquad P + P \equiv_S P. \tag{10, 11}$$

In addition, choice distributes through sequential and concurrent composition.

$$P.(Q+R) \equiv_S (P.Q)+(P.R), \tag{12}$$

$$(P+Q).R \equiv_S (P.Q)+(P.R), \tag{13}$$

$$P\|(Q+R) \equiv_S (P\|Q)+(P\|R). \tag{14}$$

In what follows, we will elude the symbol ., writing $P.Q$ simply as PQ, giving that operation highest precedence, followed by $\|$, with + having the lowest precedence. So $((P.Q)\|R)+S$ can be written as $PQ\|R+S$.

We again leave it to the reader to verify the soundness of equations (8)-(14). In order to prove soundness, we must first make a few definitions. Let a *codot* term be a term that only includes the operations . and $\|$, as well as variables; e.g., it does not contain any instances of +. A term is in *normal form* if and only if it is a codot term or the sum of codot terms such as $t_1+t_2+\cdots+t_n$.

Theorem 4.3. *If t and u are terms over the operations ., $\|$ and +, then $t \equiv_S u$ is valid if and only if $t \equiv_S u$ is provable from equations* (1)-(14) *above.*

Proof. Now every term over the operations ., $\|$, and + is equivalent to a term in normal form, by use of the distributive laws in equations (12)-(14) above. If $t \equiv_S u$, then it follows that there are terms t' and u' which are both in normal form such that the identities $t' \equiv_S t$ and $u' \equiv_S u$ are provable from Axioms (12)-(14). But then $t' \equiv_S u'$ holds. We may now write t' as $t_1+t_2+\cdots+t_m$ and u' as $u_1+u_2+\cdots+u_n$, where each t_i and u_j is a codot term. If we let $\tilde{X}=(X_1,\ldots,X_m)$ be all variables occurring in t' and u' and let $\tilde{Y}=(\{X_1\},\ldots,\{X_m\})$, we see that $t_i[\tilde{X}/\tilde{Y}]=\rho(t_i)$. Consequently,

$$\{\rho(t_1),\rho(t_2),\ldots,\rho(t_m)\} = t'[\tilde{X}/\tilde{Y}] = u'[\tilde{X}/\tilde{Y}]$$
$$= \{\rho(u_1),\rho(u_2),\ldots,\rho(u_n)\}$$

must hold since t' and u' give the same result for all instantiations, including this one. But then for each t_i there is a u_j such that $\rho(t_i)=\rho(u_j)$. From Lemma 4.1 it follows that $t_i \equiv_S u_j$. From Theorem 4.2 it then follows that $t_i \equiv_S u_j$ is provable from Axioms (1)-(7). Symmetrically, for each term u_k there is a t_l such that $u_k \equiv_S t_l$ is provable from Axioms (1)-(7). From these equations the proof of $t \equiv_S u$ is straightforward. □

5. Subsumption and the Interpolation Lemma

Consider the term $x\|y$, which is intended to represent the concurrent operation of processes x and y. In the previous section we took the point of view that the operation $\|$ *required* that x and y actually operated in parallel. Suppose we now wish a concurrency operation that does not require concurrency, but merely permits it. More specifically, we wish the meaning of $x\|y$ to allow tasks x and y to be

performed either concurrently (overlapping in time), or sequentially in either order. Then the term $x \| y$ can be seen to be in this sense a sort of generalization of the term xy. We capture the sense of this generalization in a notion called subsumption.

Let J and K be pomsets that differ only in their partial ordering, i.e., they share the same vertex set V, alphabet Σ and labelling μ, but have different partial orderings, $<_J$ and $<_K$ respectively. We say that J *subsumes* K exactly when $<_J$ is less restrictive than $<_K$, i.e., $<_J \subseteq <_K$. We use the notation $J > K$ to denote the relation "J subsumes K". The idea is that J has fewer restrictions than K, thus it is more general.

Let us consider a few examples. We assume in this paragraph that terms refer to pomsets in the obvious way. It is easy to see that $A \| B > AB$ and $A \| B > BA$ are both valid. Furthermore, $AB \| C$ subsumes ABC, ACB and CAB but does not subsume BAC, BCA, or CBA, each of which has the A and B in the wrong order. $A \| B \| C$ subsumes every pomset containing one instance of A, B and C making it a maximal element in the subsumption partial order, while strings such as ABC are the minimal elements since they subsume nothing besides themselves.

The notion of subsumption appears in Grabowski's work [9] as the "smoother" relation, i.e., he writes "J is smoother than K" where we would write "J is subsumed by K".

In this section we will investigate the properties of processes that are downwardly closed with respect to subsumption. In other words, our objects will be sets of pomsets which have the following property: if J is in the set and J subsumes K, then K is in the set. Such sets of pomsets we will refer to as *ideals* of pomsets, or merely as ideals. So for example, $\{A \| B, AB, BA\}$ and $\{AB\}$ are ideals while $\{A \| B, AB\}$ is not. Ideals obey some simple closure properties which are outlined in the following theorems.

Theorem 5.1. *The subsumption relation partially orders the class of pomsets.*

Proof. Subsumption is obviously reflexive. For transitivity, let $J \preccurlyeq K \preccurlyeq L$. Then there is a set V and a labelling μ such that $J = [V, \Sigma, <_J, \mu]$, $K = [V, \Sigma, <_K, \mu]$, and $L = [V, \Sigma, <_L, \mu]$. Furthermore, $<_L \subseteq <_K$ and $<_K \subseteq <_J$. But then it follows that $<_L \subseteq <_J$, whence $J \preccurlyeq L$. Thus subsumption is transitive. Now, suppose that $J \preccurlyeq K$ and $K \preccurlyeq J$. Then we have $<_J \subseteq <_K$ and $<_K \subseteq <_J$ giving $<_J = <_K$ and consequently, $J = K$. It follows that subsumption is antisymmetric, proving the lemma. □

Theorem 5.2. *If S and T are ideals, then so are their union, intersection, and concatenation.*

Proof. By concatenation we mean $S.T = \{JK \mid J \in S \wedge K \in T\}$. The proof is straightforward and is left to the reader. □

One difficulty that arises is that the concurrent composition of two ideals is not necessarily an ideal. However, since ideals are closed under intersection we may

define the operation ι, which takes as input a set of pomsets P and produces the *least ideal containing P*, which is defined to be the intersection of all ideals containing P. Then we define the concurrent composition of S and T as

$$S \|_I T = \iota(\{J \| K \mid J \in S \wedge K \in T\}).$$

The ι operation was first defined by Grabowski in [9] where he called it the "weakening" of a partial language. It is also closely related to Ito's $\$_n$ operator [13].

Now we have a model of the operations $.$, $\|$, and $+$ (which we take to mean union) which reflects the "permissive" view of the concurrency operation introduced in this section. Henceforth we shall refer to it as $\mathcal{M}_3$. If two terms t and r are equivalent with respect to ideals, we will say that they are I-equivalent and write $t \equiv_I r$. One reason for investigating this model is that it is closely connected to language-theoretic models of these operations, as we will see in the next section. We are already in possession of a set of equations valid in $\mathcal{M}_3$. As one might expect, the new model obeys all the equations of the old model since the objects of the new model are still sets and the operations very similar or identical. This gives us the following theorem.

Theorem 5.3. *Equations* (1)–(14) *of the previous section are valid in* $\mathcal{M}_3$ (*with* $\equiv_S$ replaced by $\equiv_I$ *and* $\|$ *interpreted as* $\|_I$).

Proof. The operations of $+$ and $.$ on ideals are identical to those operations on processes. So all we need to verify is that those axioms which involve $\|$ remain true when applied to ideals using the operation $\|_I$ for concurrent composition. Most of these are quite easily verified; we will discuss only Axioms (2) and (14). In the following, we make use of the fact that for processes P and Q, $P \| Q \subseteq P \|_I Q$.

First, we need to verify that $(P \|_I Q) \|_I R \equiv_I P \|_I (Q \|_I R)$. So let $J \in (P \|_I Q) \|_I R$. Then there are pomsets $K \in P \|_I Q$ and $L \in R$ such that $J \preccurlyeq K \| L$. Furthermore, there are pomsets $K_1 \in P$ and $K_2 \in Q$ such that $K \preccurlyeq K_1 \| K_2$, from which it follows that

$$J \preccurlyeq (K_1 \| K_2) \| L = K_1 \| (K_2 \| L).$$

But

$$K_1 \| (K_2 \| L) \in P \| (Q \| R) \subseteq P \|_I (Q \|_I R),$$

giving $(P \|_I Q) \|_I R \subseteq P \|_I (Q \|_I R)$. Symmetrically, we have

$$P \|_I (Q \|_I R) \subseteq P \|_I (Q \|_I R),$$

which gives us the desired I-equivalence.

The other I-equivalence we will verify is

$$P \|_I (Q + R) \equiv_I (P \|_I Q) + (P \|_I R).$$

Suppose that $J \in P \|_I (Q + R)$. Then there are pomsets $K \in P$ and $L \in Q + R$ such that $J \preccurlyeq K \| L$. Without loss of generality, assume that $L \in Q$. Then

$$K \| L \in P \| Q \subseteq P \|_I Q + P \|_I R,$$

giving $J \in P \|_1 Q + P \|_1 R$. Now suppose that $J \in P \|_1 Q + P \|_1 R$. Without loss of generality, assume that $J \in P \|_1 Q$. Then there exist $K \in P$ and $L \in Q$ such that $J \leqslant K \| L$. But

$$K \| L \in P \|(Q+R) \subseteq P \|_1 (Q+R),$$

hence $J \in P \|_1 (Q+R)$. This proves the desired I-equivalence, ending the proof. □

But what about completeness? There is exactly one more equation which, when added to equations (1)–(14), gives a complete axiomatization of the model $\mathcal{M}_s$. We call it the subsumption axiom since it encapsulates everything there is to know about the subsumption relation. Before stating the axiom, we define a bit of notation. In general, we will write $A \leqslant B$ to denote $A + B = B$. In particular $A \leqslant_1 B$ denotes $A + B \equiv_1 B$. The notion is that since + denotes set union, $\leqslant$ denotes set containment. The definition of $\leqslant$ justifies our calling the subsumption axiom an equation, even though it appears to be an inequality:

$$(x \|_1 y)(u \|_1 v) \leqslant_1 xu \|_1 yv. \tag{15}$$

Substituting for the variables of equation (15) gives us various interesting laws including

$$xv \leqslant_1 x \|_1 v \quad \text{and} \quad x(u \|_1 v) \leqslant_1 xu \|_1 v.$$

Validity of the subsumption axiom is based upon the following theorem about subsumption and pomset-definable operations on ideals. Before we can state the theorem, we must define just what we mean by a pomset-definable operation on ideals. Let J be a pomset. It then defines an operation on pomsets and indeed on sets of pomsets, as explained in Section 2. However, if we let the arguments to that operation be ideals, we have no guarantee that the result will be an ideal. The solution is to define the operation on *ideals* defined by J to be $\iota(J[g])$ for any assignment g that maps symbols to ideals. In other words, we use the operation defined by J on sets of pomsets and take the least ideal containing the result. Now we can state our theorem that makes subsumption so interesting.

Theorem 5.4. *Let J and K be pomsets. Then $J \geqslant K$ if and only if, for all ideal assignments g, $\iota(J[g])$ contains $\iota(K[g])$.*

Proof. First, since J subsumes K, they have (or can be thought of having) the same vertex set, and the same labelling. So when we compare $J[g]$ with $K[g]$, we find that, for any pomset K' in $K[g]$, there is a corresponding pomset J' in $J[g]$ that subsumes K'. We construct J' by substituting the exact same pomsets for the vertices of J as were substituted for those vertices in K to form K'. Now J' subsumes K', hence K' is an element of $\iota(J[g])$, so $\iota(J[g]) \supseteq K[g]$. Because $\iota(K[g])$ is the smallest ideal containing $K(g)$, we obtain $\iota(J[g]) \supseteq \iota(K[g])$.

Next, assume $\iota(J[g])$ contains $\iota(K[g])$ for all ideal assignments g. Then if we let g be the identity assignment, we have $\iota(J) \supseteq \iota(K)$. But this implies that $K \in \iota(J)$, which implies $J \succcurlyeq K$. □

Corollary 5.5. *If t_1 and t_2 are codot terms, then $t_1 \leqslant_1 t_2$ if and only if $\rho(t_1) \leqslant \rho(t_2)$.*

Proof. Let h map each variable x to the trivial pomset ideal $\{x\}$. The result follows immediately. □

This theorem gives us a criterion for deciding whether two terms t_1 and t_2 which do not contain a + operation obey the formula $t_1 \leqslant_1 t_2$. Remembering the natural correspondence of such terms with series-parallel pomset-definable operations, we check to see whether $\rho(t_1)$ is subsumed by $\rho(t_2)$, that is, whether, as *pomsets*, t_2 subsumes t_1. This condition is in fact equivalent to $t_1 \leqslant_1 t_2$ by the above theorem. So we break our proof of the completeness of Axioms (1)–(15) into two parts. The first demonstrates that formulas of the form $t_1 \leqslant_1 t_2$, where t_1 and t_2 are codot terms, can be proved from the axioms, and the second generalizes the result to terms of all sorts.

Completeness of the axioms for codot terms rests on the following lemma, which we refer to as the Interpolation Lemma.

Lemma 5.6 (Interpolation Lemma). *If t_1 and t_2 are codot terms that obey $t_1 \leqslant_1 t_2$, then either $t_1 \equiv_1 t_2$ or there is another codot term t_3 such that $t_1 < t_3 \leqslant_1 t_2$ holds and the formula $t_1 <_1 t_3$ is provable from Axioms* (1)–(15).

Proof. If $t_1 \equiv_1 t_2$, then it follows that $\rho(t_1) = \rho(t_2)$, by taking the trivial substitution $g(x) = \{x\}$ in $t_1[g] = t_2[g]$.

Otherwise we have $\iota(\rho(t_1)) \subset \iota(\rho(t_2))$ which implies $\rho(t_1) < \rho(t_2)$ by Theorem 5.4. By the definition of subsumption, we may assume that $V_{\rho(t_1)}$ equals $V_{\rho(t_2)}$, which we will henceforth call V, and that $\mu_{\rho(t_1)} = \mu_{\rho(t_2)}$, which we will refer to merely as μ. Since the subsumption is strict, there exist two vertices $x, y \in V$ such that $x <_{\rho(t_1)} y$, but x and y are incomparable under $<_{\rho(t_2)}$.

Let H be a smallest (in number of vertices) pomset satisfying the following conditions:

(1) There exist terms r and t' such that $t' \equiv_1 t_1$, r is a subterm of t' and $H = \rho(r)$.

(2) There exist u and v in V_H such that $u <_{\rho(t_1)} v$ but $u \not<_{\rho(t_2)} v$.

Since $\rho(t_1)$ satisfies the above conditions, we know that such an H must exist. Because of the minimality condition, H must be of the form $H_1.H_2$, where neither H_1 nor H_2 is the result of a concatenation.

We will find a term s such that $r \leqslant_1 s$ is provable and such that replacing H by $\rho(s)$ in $\rho(t_1)$ results in a pomset J such that $J \not\succcurlyeq \rho(t_2)$. Since monotonicity of . and $\|$ is provable, we know that replacing r with s in t' will result in the promised t_3 in the statement of the lemma.

Now H_1 and H_2 consist of some number of connected components, $F_1, \ldots, F_m$ and $G_1, \ldots, G_n$ respectively. Observe that if $m = 1$, then F_1 consists of exactly one vertex, by minimality of H. A similar argument holds for n and G_1.

Observe that $\rho(t_1)$ and $\rho(t_2)$ are identical when restricted to V_{F_i} or V_{G_j} for all i, j such that $1 \leq i \leq m$, $1 \leq j \leq n$. For this reason we will not distinguish between them. We will say the F_i is *connected* to G_j iff there are vertices $x \in V_{F_i}$ and $y \in V_{G_j}$ with $x <_{\rho(t_2)} y$.

Claim. *There exist i and j such that F_i and G_j are not connected.*

Proof. Without loss of generality assume $u \in V_{F_1}$, $v \in V_{G_1}$, and $u \not<_{\rho(t_2)} v$. Then there is a maximal element w of F_1 and a minimal element w' of G_1 such that $w \not<_{\rho(t_2)} w'$. Otherwise we would have $u <_{\rho(t_2)} v$ by transitivity.

If $m = n = 1$, then F_1 and G_1 each consist of a single vertex, and are not connected, hence satisfying the claim.

If $m = 1$ and $n > 1$, then F_1 consists of the single vertex u. If F_1 is not connected to G_1, we are done. Otherwise, it follows that $d(w', u) > 1$. If F_1 is not connected to G_2, we are done. Otherwise, since G_1 and G_2 are not connected, $d(w', y) > 2$ for all $y \in V_{G_2}$. This contradicts N-freeness of $\rho(t_2)$, so we conclude that either G_1 or G_2 must not be connected to F_1. A symmetric argument holds when $m > 1$ and $n = 1$.

Now suppose that $m > 1$ and $n > 1$. Consider F_1, F_2, G_1, and G_2. If either of F_1 or F_2 is not connected to either of G_1 or G_2, then we are done. So we assume that each of the possible four connections exist. Let z be a vertex of F_2. Since $\rho(t_2)$ is connected when restricted to $V_{F_1} \cup V_{F_2} \cup V_{G_2}$, it follows that w and z have an upper bound in G_2, call it x'. Likewise it follows that w' and x' have a lower bound x in F_2. Then $w < x' > x < w'$ by construction. Furthermore, w and x are unrelated, coming from different F_i's, as are w' and x'. But $w \not< w'$ from above, violating N-freeness of $\rho(t_2)$ (see Fig. 3). So we must conclude that an F_i is not connected to a G_j. This proves the claim. □

Without loss of generality, we say that F_1 is not connected to G_1. Suppose that F_1 is not connected to any G_j. Then a term of the form

$$F_1 \| ((F_2 \| \cdots \| F_m).(G_1 \| \cdots \| G_n))$$

will suffice for s, because of the equation $(x \| y)z \leq_1 x \| yz$ which follows from the subsumption axiom by

$$(x \| y)z \equiv_1 (x \| y)(1 \| z) \leq_1 x.1 \| yz \equiv_1 x \| yz.$$

Therefore, in what follows we assume that F_1 is connected to some G_j.

We wish to partition the F_i's and the G_j's into two sets each, called P_1, P_2, Q_1 and Q_2 respectively, such that no element of P_1 is connected to any element of Q_2 (via $<_{\rho(t_2)}$), and likewise for P_2 and Q_1. Furthermore we want P_1, Q_1 and Q_2 to be nonempty.

We form P_1, P_2, Q_1, and Q_2 as follows: Let Q_1 contain every G_j that is connected to F_1. Then let P_1 contain all F_i such that F_i is connected to some element of Q_1.

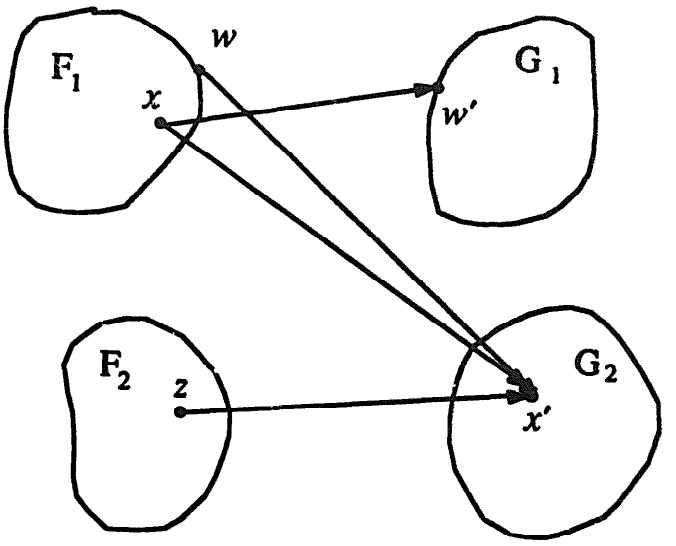

Fig. 3. w, x', x, and w' form an "N".

Next, let P_2 be the remaining F_i's and Q_2 the remaining G_j's. Since we assumed that F_1 was connected to some G_j, it follows that P_1 and Q_1 are nonempty. Since G_1 is not connected to F_1, it follows that Q_2 is nonempty. By construction, there can be no connections between Q_1 and P_2, so all that remains to be shown is that there are no connections between P_1 and Q_2.

Suppose that there is an $F_i \in P_1$ and a $G_j \in Q_2$ which are connected. Now F_1 cannot be directly connected to G_j or else we would have $G_j \in Q_1$. There must be a $G_k \in Q_1$ directly connected with both F_1 and F_i but then it follows that, for all $x \in F_1$ and $y \in G_j$, $d(x, y) \geq 3$ which contradicts N-freeness of $\rho(t_2)$.

So we have formed P_1, P_2, Q_1, and Q_2 as desired. We now note that there are terms $f_1, \ldots, f_m$ and $g_1, \ldots, g_n$ and integers k and l such that

$$P_1 = \{\rho(f_i) \mid 1 \leq i \leq k\}, \qquad P_2 = \{\rho(f_i) \mid k+1 \leq i \leq m\},$$

$$Q_1 = \{\rho(g_j) \mid 1 \leq j \leq l\}, \qquad Q_1 = \{\rho(g_j) \mid l+1 \leq j \leq n\}.$$

But now we see that the term

$$((f_1 \| \cdots \| f_k).(g_1 \| \cdots \| g_l)) \| ((f_{k+1} \| \cdots \| f_m).(g_{l+1} \| \cdots \| g_n))$$

will suffice for s because of the axiom $(x \| y)(u \| v) \leq_I xu \| yv$ and because we may write

$$H_1 = \rho(f_1 \| \cdots \| f_m), \qquad H_2 = \rho(g_1 \| \cdots \| g_n).$$

In every case the number of edges in $\rho(s)$ is smaller than the number of edges in $\rho(r)$, making the subsumption strict and proving the lemma. □

The Interpolation Lemma can be used to give a proof of a valid formula $t_1 \leq_I t_2$ as follows. Repeatedly find "interpolating" terms as given by the lemma so that we

have a chain of inequalities

$$t_1 <_1 s_1 <_1 s_2 <_1 \cdots <_1 s_m \equiv_1 t_2$$

each step of which is provable from the axioms. We may be sure that such a chain is finite since the total number of pomsets on any finite multiset is finite. Thus we have a proof that $t_1 \leqslant_1 t_2$.

All that remains is to show how to use these results when proving valid formulas involving the operation $+$. To do this we again make use of the normal form for terms in our algebra. Since any term is provably equivalent to a codot term, what we must show is that the axioms allow us to prove all valid equations of the form $t \equiv_1 u$ such that t and u are in normal form. Our plan of action will be to further reduce the problem to proving valid *inequalities* between *codot* terms. Reducing the problem to inequalities is not difficult since $A \equiv_1 B$ is provably equivalent to $A \leqslant_1 B \leqslant_1 A$, because of idempotence of choice (Axiom (11)) and transitivity of equality. Reducing the problem to one involving only codot formulas is slightly more complicated and rests upon the next two lemmas, one of which is syntactic, the other semantic. First, the syntactic lemma.

Lemma 5.7. *For all i and j such that $1 \leqslant_1 i \leqslant_1 m$ and $1 \leqslant_1 j \leqslant_1 n$, let t_i and r_j be codot terms. Then the inequality $t_1 + \cdots + t_m \leqslant_1 r_1 + \cdots + r_n$ can be proved from Axioms (1)-(15) if and only if, for all i such that $1 \leqslant_1 m$, the inequality $t_i \leqslant_1 r_1 + \cdots + r_n$ can be proved from the axioms.*

Proof. Let T and R denote the terms $t_1 + \cdots + t_m$ and $r_1 + \cdots + r_n$ respectively. Now suppose that $T \leqslant_1 R$ can be proved from the axioms. Observe that since

$$\vdash t_1 + T \equiv_1 t_1 + (t_1 + t_2 + \cdots + t_m) \equiv_1 t_1 + (t_2 + \cdots + t_m) \equiv_1 T,$$

we have $\vdash t_1 \leqslant_1 T$. But $\vdash T \leqslant_1 R$ by assumption, hence $\vdash t_1 \leqslant_1 R$ follows from transitivity. This can be repeated for each t_i.

Now assume that $t_i \leqslant_1 R$ can be proved for each i between 1 and m. Then it follows, for example, that $\vdash t_1 + t_2 \leqslant_1 R$ because

$$\vdash t_1 + t_2 + R \equiv_1 t_1 + (t_2 + R) \equiv_1 t_1 + R \equiv_1 R$$

follows from repeated use of the definition of $\leqslant_1$. This reasoning can be used inductively to show that $\vdash T \leqslant_1 R$. □

The above proof did not really depend on the fact that the terms in question were codot terms, nor did it depend on the particular model $\mathcal{M}_s$ that we have in mind for the terms. The next lemma depends vitally upon both of these facts.

Lemma 5.8. *If t is a codot term and r_j is a codot term for all j between 1 and n, then the inequality $t \leqslant_1 r_1 + \cdots + r_n$ is valid if and only if there exists some k between 1 and n such that $t \leqslant_1 r_k$ is valid.*

Proof. This lemma rests on the fact that in the model in question ($\mathcal{M}_s$), the + operation is interpreted as set union and the $\leqslant_1$ relation turns out to be set containment. The point is that the least ideal containing $A_1 \cup \cdots \cup A_n$ is in fact $A_1 \cup \cdots \cup A_n$. Now if the ideal $\iota(\rho(t))$ is a *subset* of $A_1 \cup \cdots \cup A_n$, then it follows that the pomset $\rho(t)$ is an *element* of $A_1 \cup \cdots \cup A_n$. But then $\rho(t)$ must either be an element of A_i for some i. Since A_i is an ideal, it must contain all pomsets subsumed by $\rho(t)$, so that $\iota(\rho(t))$ must be a subset of A_i. Letting $A_i = \iota(\rho(r_i))$ we obtain the desired result.

The reverse direction is easy and is left to the reader. □

In the above lemma, no mention is made of provability via the axioms. However, we observe that if the formula $t \leqslant_1 r_k$ given by the above theorem is provable from Axioms (1)–(15), then so is $t \leqslant_1 r_1 + \cdots + r_n$ by simple transitivity.

Now we have completed the trek; we can now prove any equation valid in the model $\mathcal{M}_s$. This is done by first finding equivalent inequalities and converting each side to normal form. Then from these inequalities using the above lemma we can find a finite set of codot inequalities, with but a single codot term on each side, which are valid and from which the original inequalities can be proved. Finally, we observe that a codot inequality can be proved using the Interpolation Lemma. So we may sum up with the following two theorems.

Theorem 5.9. *Axioms* (1)–(15) *are complete for the model* $\mathcal{M}_s$.

Theorem 5.10. *The theory of pomset ideals over* ., ||, *and* + *is decidable.*

Proof. Determining whether $t \leqslant_1 r$ for codot terms t and r is equivalent to deciding whether $\rho(t)$ is subsumed by $\rho(r)$, which is just a special case of subgraph isomorphism. This allows us to find the r_k's in the above lemma algorithmically by exhaustive search. The method in the interpolation lemma is also algorithmic because of the finiteness of the sets involved. (The set of all pomsets over some fixed finite multiset of symbols is finite.) All other calculation are syntactic in nature and easily carried out by computer. A detailed analysis of the running time of such an algorithm is beyond the scope of this paper. A quick calculation reveals it to be at least doubly exponential, but there is certainly room for improvement. □

6. Closure properties

It is well-known that the equational theory of languages is not finitely axiomatizable when the Kleene star operation, or concatenation closure, is added [25]. What about the equational theory of processes? As we will see in the next section, the equational theory of processes under the regular operations of +, ., and * is the same as the equational theory of languages under those operations. Therefore, the

theory of processes under the regular operations is not finitely axiomatizable. But what happens if we add $\|$ and its closure †? It is perhaps not too surprising that the result is the same: The resulting theory is not finitely axiomatizable.

In order to show this, we will define, for each prime p, an algebra A_p that is designed to violate a particular kind of axiom obeyed by languages and pomsets alike. However, this construction will have the property that any *finite* set of axioms will be satisfied in some A_p.

Consider the algebra $A = (\{a\}^*, 1, ., \|)$ of all words over the single symbol a, where the operations . and $\|$, as well as the constant 1, are interpreted as the standard language operations. Note that the identity $yz \equiv y\|z$ holds in this algebra as well as all of the usual identities for S-equivalence. Now define $y \equiv_p z$ to hold if and only if $y = a^m$, $z = a^n$ and $m \equiv n \pmod p$, taking $a^0 = 1$.

Lemma 6.1. *The relation $\equiv_p$ is a congruence on A.*

Proof. Left to reader. □

For a given p, we may reduce A by this congruence and call the resulting quotient Q_p. We observe in passing that Q_p has p elements.

Now we form the power set 2^{Q_p} and extend the operations . and $\|$ to it, pointwise in the obvious fashion. We add the operation + which is interpreted as set union, and the constant symbol 0 which is interpreted as the empty set, and call the result P_p. The upper bound of all the elements of P_p is the set $\{1, a, aa, \ldots, a^{p-1}\}$ which we will refer to as the *top* of P_p and denote by T_p.

We now adjoin a new element α^* to P_p, which obeys the identities

$$\alpha^* x = x\alpha^* = \alpha^* \| x = \alpha^* + x = \alpha^* \quad \text{for all } x \neq 0.$$

Next we define $x^\dagger$ and x^* by $0^\dagger = 0^* = 1^\dagger = 1^* = 1$ and $x^\dagger = x^* = \alpha^*$ for all other $x \in P_p$. This defines the desired algebra A_p. Finally, we define the relation $\cong$, a subset of $A_p \times A_p$, to be the same as the identity relation except that $\alpha^* \cong T_p$.

Lemma 6.2. *The relation $\cong$ is a congruence on A_p. The quotient $A_p/\cong$, when restricted to the operations +, ., and $\|$, is isomorphic to P_p.*

Proof. Clearly, $\cong$ is an equivalence relation. By definition, $T_p^\dagger = T_p^* = \alpha^*$. Since in A_p the operations . and $\|$ coincide, they must also coincide in the quotient of A_p by $\cong$, hence it suffices to show that $T_p \| y = T_p + y = T_p$ for all $y \neq 0$. However, $A_p/\cong$ is isomorphic to the powerset of the integers modulo p where $\|$ and . are interpreted as pointwise addition mod p. The lemma then follows from elementary number theory. □

Let Γ_S denote the set of all identities for S-equivalence, i.e., the set of all equations implied by Axioms (1)-(14) of the previous section.

Lemma 6.3. *For all primes p,* $(A_p/\cong) \models \Gamma_S$.

Proof. Our aim is to show that every identity for S-equivalence holds in $A_p/\cong$. Since equations are preserved by homomorphisms, we will proceed by showing that $A_p/\cong$ is the image of a particular algebra of sets of pomsets under a certain homomorphism h. Thus all the identities that hold for pomset ideals can be seen to hold in $A_p/\cong$ as well.

Let A be an algebra whose elements are sets of finite pomsets, each of which is labelled only with the symbol a. Clearly, $A \models \Gamma_S$. We now proceed to define, for each prime p the mapping $h_p : A \to A_p/\cong$. If x is an element of A and $J \in x$, then let $h_p(J) = a^k$, where $0 \leq k < p$ and $k \equiv |J| (\mathrm{mod}\, p)$. Then define $h_p(x) = \{h_p(J) \mid J \in x\}$. The reader may easily verify that h_p is onto, and that

$$h_p(x+y) = h_p(x) + h_p(y), \qquad h_p(xy) = h_p(x)h_p(y),$$
$$h_p(x \| y) = h_p(x) \| h_p(y).$$

Now if x contains a pomset J such that $|J| \not\equiv 0 (\mathrm{mod}\, p)$, it follows that $h_p(x^*) = h_p(x)^* = T_p$ since $|J|$ and p are relatively prime. Otherwise $h_p(x^*) = a^0 = 1 = h_p(x)^*$. Similarly, $h_p(x^\dagger) = h_p(x)^\dagger$. Thus we conclude that h_p is a homomorphism, proving the lemma. □

Next we present the equations which are S-equivalences but do not hold in A_p. Let $x^{<n}$ denote the term $1 + x + \cdots + x^{n-1}$.

Lemma 6.4. *For all* $n > 0$, $x^* \equiv_S (x^n)^* x^{<n}$.

Proof. The equations in question are known to hold for languages [6], hence they must hold for ideals by Theorem 7.1 of the next section. □

Proposition 6.5. *For all primes p, the algebra* A_p *is not a model of* $x^* = (x^n)^* x^{<n}$.

Proof. In A_p the right-hand side of the equation evaluates to T_p while the left-hand side evaluates to a^* since $x^p = 1$. □

All that remains to be shown is that, for any finite subset Σ of Γ_S, there is a p such that $A_p \models \Sigma$. To show this, we must first introduce the notion of the length of a formula. Noting that

$$(x+y)^* \equiv_S (x^* y^*)^* \quad \text{and} \quad (x+y)^\dagger \equiv_S (x^\dagger \| y^\dagger),$$

we see that it is possible to extend the notion of normal form defined in Section 2. We will say that a term is *reduced* just when it is 0, 1, or some variable a, or when it is of the form xy, $x \| y$, x^*, or $x^\dagger$, where x and y are reduced terms. A term is in *extended normal form* when it is the sum of reduced terms. The *length* of a term t is the smallest number of reduced terms appearing in any term in extended normal form which is S-equivalent to t. The length of an equation $t = r$ is the maximum of the lengths of t and r. We may now show the main result of this section.

Theorem 6.6. *For every prime p, every identity of Γ_S of length less than p holds in A_p.*

Proof. Suppose that t and r are terms over the variables $\tilde{X} = (X_1, \ldots, X_m)$ such that $t \equiv_S r$ but that there exists an $\tilde{x} = (x_1, \ldots, x_m)$ in A_p^m such that $t[\tilde{X}/\tilde{x}] \neq r[\tilde{X}/\tilde{x}]$. Since $A_p/\cong$ is a model of Γ_S, it must be the case that one side of this identity evaluates to T_p, while the other side evaluates to α^*. Without loss of generality, assume that $t[\tilde{X}/\tilde{x}] = T_p$ and $r[\tilde{X}/\tilde{x}] = \alpha^*$.

We claim that every variable appearing inside a dagger or star in t must also appear inside a dagger or star (not necessarily respectively) in r and vice versa. Letting $\tilde{Y} = (\{X_1\}, \ldots, \{X_m\})$ we observe that any S-equivalence must hold under the substitution $\tilde{X}/\tilde{Y}$, which suffices to verify the claim.

Now replace every variable of t and r which does not appear under a star or dagger by 1. This results in an equation $t' \equiv_S r'$ which is no longer than the original equation. Furthermore, $T_p = t'[\tilde{X}/\tilde{x}] \neq r'[\tilde{X}/\tilde{x}] = \alpha^*$.

Let X_i be a variable which appears in t'. Then x_i must be of the form $\{a^k\}$. For if x_i contained strings a^k and a^l such that $k \not\equiv l \pmod{p}$, we would then have a nonzero power of a appearing under a dagger or star in t', resulting in $t'[\tilde{X}/\tilde{x}]$ evaluating to α^* rather than to T_p.

But if every x_i is of the form $\{a^k\}$, then each reduced term on the left side yields an element of the form $\{a^j\}$, i.e., it only contains one string. We conclude that the length of t' is at least p since $t'[\tilde{X}/\tilde{x}] = T_p = \{a^0, a^1, \ldots, a^{p-1}\}$. Hence also the length of t and thence the length of the identity $t = r$ is at least p. □

Corollary 6.7. *The equational theory of processes under the operations ., ||, +, *, and $^+$ is not finitely axiomatizable.*

Proof. Let Γ_0 be a finite set of equations, intended to axiomatize S-equivalence. Let m be the maximum of the lengths of the equations in Γ_0. Choose p to be a prime larger than m. Then we see that $A_p \models \Gamma_0$ but $A_p \not\models \Gamma_S$. Hence Γ_0 does not axiomatize S-equivalence. □

7. Connections with language theory

In this section we will survey results similar to those of the previous section which apply to a language-theoretic model, which is in many ways the "standard" model of the operations ., ||, and +. In this model variables are interpreted as languages, i.e., as sets of strings or tomsets. The operations . and + are interpreted as the well-known and well-understood operations of language concatenation and set union. The interpretation of || is that of the less well-known and certainly less understood operation of interleaving or *shuffling*. This operation is defined by the following equation:

$$L_1 \| L_2 = \{x_1 y_1 x_2 y_2 \ldots x_n y_n \mid x_1 x_2 \ldots x_n \in L_1 \wedge y_1 y_2 \ldots y_n \in L_2\}.$$

We will say that two terms t and r are L-equivalent ($t \equiv_L r$) if and only if t and r denote the same language whenever their variables are interpreted as languages. First, we observe that Axioms (1)–(15) hold for L-equivalence as well. We demonstrate this fact by introducing a new operation λ which maps pomsets to languages. We call λ the *linearization operator*. Specifically, if J is a pomset, then $\lambda(J)$ consists of exactly those tomsets (strings) that are subsumed by J. Thus $\lambda(J)$ is a subset of $\iota(J)$. The linearization operator was first introduced by Grabowski in [9], who called it the *smoothing* of a partial string.

Grabowski showed, and we leave it to the reader as an easy exercise, that λ is a homomorphism mapping the ideal model to the language model (or mapping the sets of pomsets model to the language model); the relevant equations are

$$\lambda(xy)=\lambda(x)\lambda(y), \qquad \lambda(x\|y)=\lambda(x)\|\lambda(y), \qquad \lambda(x+y)=\lambda(x)+\lambda(y),$$

$$\lambda(x^*)=\lambda(x)^* \quad \text{and} \quad \lambda(x^\dagger)=\lambda(x)^\dagger.$$

From this it is an easy step to see that Axioms (1)–(15) must hold for the language model since valid equations are always preserved by homomorphisms.

Theorem 7.1. *Let t and r' be terms over $.$, $\|$, $+$, *, and †. Then $t \equiv_I r$ implies $t \equiv_L r$.*

Proof. Trivial since every language is an ideal. □

What about the converse? Are there equations that hold for languages but not for ideals? If we restrict our attention to the operations of $+$, $.$, and *, then the answer is no, because of the following theorem.

Theorem 7.2. *Every algebra of processes over the operations $+$, $.$ and * is isomorphic to an algebra of languages over those operations.*

Proof. Let $\mathcal{P}$ be an algebra of processes as in the statement of the theorem. Then let Σ be the set of all prime factors of pomsets found in $\mathcal{P}$. We shall treat the elements of Σ as symbols of an alphabet, and the language algebra $\mathcal{L}$ we construct will have Σ as its alphabet.

Let h' be the unique mapping from pomsets to Σ^* obeying $h'(1)=\varepsilon$, $h'(J)=J$ for all prime J, and $h'(J.K)=h'(J)h'(K)$. Since factorization into primes disregarding associativity is unique, such an h' must exist. Furthermore, it is one-to-one when restricted to Σ.

We may now define $h: \mathcal{P} \to 2^{\Sigma^*}$ by letting, for any process P, $h(P)=\{h'(J) \mid J \in P\}$. The desired language algebra $\mathcal{L}$ is the image of $\mathcal{P}$ under h.

We claim that h is a homomorphism. $h(P.Q)=h(P)h(Q)$ follows from the definition of h'. Furthermore, $h(\sum P_i)=\sum h(P_i)$ follows from the definition of h. But then it follows that

$$h(P^*)=h(1+P+PP+\cdots)=h(1)+h(P)+h(P)h(P)+\cdots=h(P)^*.$$

So h is a homomorphism. To complete the proof, we observe that h' is one-to-one on the pomsets which appear in $\mathcal{P}$, hence h is also one-to-one. □

So we see that if we do not make use of the concurrency operations, languages have the same equational theory as processes. When we add the concurrency operations, things are not so simple. In particular, it is not the case that Axioms (1)–(15) are complete for the language model. There is at least one other equation which holds in this model, to wit

$$(xy)\|(xy) \leqslant_L x\|(x(y\|y)). \tag{16}$$

Observe that this equation implicitly contains the operation + because of the definition of $\leqslant_L$.

Proposition 7.3. *Equation* (16) *is valid for L-equivalence, but not for S-equivalence or I-equivalence.*

Proof. We give an informal justification of the validity of equation (16). Details are left to the reader.

The terms on each side of the equation involve two variables x and y. In the language model these variables will be interpreted as languages, call them L_x and L_y respectively. These substitutions result in languages corresponding to the left-hand and right-hand sides of the equation, call them L_ℓ and L_r respectively. Now any string z in L_ℓ must begin with an interleaving of strings from L_x, call them z_1 and z_2. At some point, we reach the end of one of these strings, call it z_1 and (perhaps later) begin interleaving a string from L_y. Now to see that z is in L_r, we only need to observe that we can always regard z as being the contribution of the second instance of x in the right-hand term (i.e., the boldfaced x in $x\|(x(y\|y))$). Then the strings contributed by the two instances of L_y can be interleaved to form z without any further constraint.

Invalidity of (16) in the pomset model follows immediately from the fact that $\rho((xy)\|(xy)) \not\leqslant \rho(x\|(x(y\|y)))$. □

Although there are many results about the shuffle operation when applied in a regular expression setting, there are very few that apply to our algebraic setting. The equational theory of languages under $\cdot$, $\|$, and + has thus far resisted characterization. Indeed, whether the equational theory of languages under just concatenation and shuffle is finitely axiomatizable is not known. We conjecture that Axioms (1)–(7) of Section 4 are complete for this algebra as well.

8. Conclusions and open problems

At the time of this writing there are a multiplicity of formalisms and description languages for parallel computation. It is naturally of interest which formalisms are

equivalent to others in expressive power. However, there is some difficulty in obtaining results of this type since there is no general agreement on what basis such a comparison is to be made. An analogy may be made to the early work on computability, where many different models of computation such as Turing machines, Church's lambda calculus, etc. were proved to be equivalent in the sense that the set of functions that were realizable on each of these models turned out to be the same. It would be nice if a similar unification could be made in the field of concurrent computation. Winskel has carried out some important work in this area in [31].

In order to carry out such a comparison in the theory of concurrent computation, it will be necessary to find a formal object that can play the role that functions did for researchers in the theory of computation. They could say that two programs were equivalent when they computed the same function from inputs to outputs (side effects are considered part of the output). This does not tell the whole story for semantics of programming languages, but that is a different story. Now we would like something to play this part for processes. Pratt has made several arguments for the use of pomsets in [24], where he also describes how Petri nets can be modelled as pomsets. While pomsets are by no means the "last word" on this subject, what does seem clear is that we must move toward a model that does not depend on interleaving, such as the shuffle operation does.

Our work shows that the language model, which depends on interleaving, is less general (there are more axioms) and more difficult to obtain results about than the pomset model. We would argue that any model that is so difficult to prove things about is also difficult to understand well enough to use as a foundation for concurrent programming. In addition, a model that depends on interleaving seems bound to identify objects that are kept distinct by the pomset model. Just exactly how useful and important it is to distinguish between these objects is not clear, and merits further study.

References

[1] K. Abrahamson, Modal logic of concurrent nondeterministic programs, in: *Proc. Symp. on Semantics of Concurrent Computation*, Lecture Notes in Computer Science **70** (Springer, Berlin, 1979) 21-33.

[2] T. Araki, T. Kagimasa and N. Tokura, Relations of flow languages to Petri net languages, Programming Languages Group Memo No. 79-01, Department of Information and Computer Sciences, Osaka University, 1979.

[3] W. Brauer, ed., *Net Theory and Applications*, Lecture Notes in Computer Science **84** (Springer, Berlin, 1980).

[4] J.D. Brock and W.B. Ackerman, Scenarios: a model of nondeterministic computation, in: J. Diaz and I. Ramos, eds., *Formalization of Programming Concepts*, Lecture Notes in Computer Science **107** (Springer, Berlin, 1981) 252-259.

[5] S.D. Brookes, C.A.R. Hoare and A.W. Roscoe, A theory of communicating sequential processes, *J. ACM* **31**(3) (1984) 560-599.

[6] J.H. Conway, *Regular Algebra and Finite Machines* (Chapman & Hall, London, 1971).

[7] J.L. Gischer, Shuffle languages, Petri nets, and context-sensitive languages, *Comm. ACM* 24(9) (1981) 597-605.
[8] J.L. Gischer, Partial orders and the axiomatic theory of shuffle, Tech. Rept. No. STAN-CS-84-7033, Stanford University, 1984.
[9] J. Grabowski, On partial languages, *Ann. Soc. Math. Polon. Ser. IV. Fund. Math.* IV(2) (1981) 427-498.
[10] I. Greif, Semantics of communicating parallel processes, Ph.D. Thesis, Massachusetts Institute of Technology (Project MAC TR-154), 1975.
[11] C.A.R. Hoare, *Communicating Sequential Processes* (Prentice-Hall, Englewood Cliffs, NJ, 1985).
[12] T. Ito and S. Ando, A complete axiom system of super-regular expressions, in: J.L. Rosenfeld, ed., *Proc. IFIP 74* (North-Holland, Amsterdam, 1974) 661-665.
[13] T. Ito, On behaviors of parallel processes with duration, in: *Proc. Conf. on Information Sciences and Systems*, Johns Hopkins University, Baltimore (1985).
[14] M. Jantzen, The power of synchronizing operations on strings, *Theoret. Comput. Sci.* 14(2) (1981) 127-154.
[15] M. Jantzen, Extending regular expressions with iterated shuffle, *Theoret. Comput. Sci.* 38 (1985) 223-247.
[16] G. Kahn and D.B. MacQueen, Coroutines and networks of parallel processes, in: B. Gilchrist, ed., *Proc. IFIP 77* (North-Holland, Amsterdam, 1977) 993-998.
[17] T. Kimura, An algebraic system for process structuring and interprocess communication, in: *Proc. 8th Ann. Symp. on Theory of Computing* (1976) 92-100.
[18] L. Lamport, Time, clocks, and the ordering of events in a distributed system, *Comm. ACM* 21(7) (1978) 558-564.
[19] L. Lamport, The mutual exclusion problem: Part I—A theory of interprocess communication, *J. ACM* 33(2) (1986) 313-326.
[20] R. Milner, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science 92 (Springer, Berlin, 1980)
[21] R. Milner, A complete inference system for a class of regular behaviours, *J. Comput. System Sci.* 28 (1984) 439-466.
[22] V.R. Pratt, On the composition of processes, in: *Proc. 9th Ann. Symp. on Principles of Programming Languages* (1982) 213-223.
[23] V.R. Pratt, Some constructions for order-theoretic models of concurrency, in: *Proc. Conf. on Logics of Programs*, Lecture Notes in Computer Science 193 (Springer, Berlin, 1985) 267-283.
[24] V.R. Pratt, Modelling concurrency with partial orders, *Internat. J. Parallel Programming* 15 (1987) 33-71.
[25] V.N. Redko, On definining relations for the algebra of regular events, *Ukrain. Mat. Ž.* 16 (1964) 120-126; in Russian.
[26] A. Salomaa, Two complete axiom systems for the algebra of regular events, *J. ACM* 13(1) (1966) 158-169.
[27] A.C. Shaw, Software description with flow expressions, *IEEE Trans. Software Engineering* SE-4(3) (1978) 242-254.
[28] G. Slutzki, Descriptional complexity of concurrent processes, in: Lecture Notes in Computer Science 88 (Springer, Berlin, 1980) 601-611.
[29] J. Valdes, R.E. Tarjan and E.L. Lawler, The recognition of series parallel digraphs, *SIAM J. Comput.* 11(2) (1981) 298-313.
[30] G. Winskel, Events in computation, Ph.D. Thesis, University of Edinburgh, 1980.
[31] G. Winskel, Categories of models for concurrency, in: Lecture Notes in Computer Science 197 (Springer, Berlin, 1985) 246-267.
[32] R.L. Graham, D.L. Knuth and T.S. Motzkin, Complements and transitive closures, Tech. Rept. No. STAN-CS-71-214, Stanford University, 1971.